Obrada slike i računalni vid

Projektni zadatak

Uklanjanje šuma pomoću DFT i FFT

Stjepan Salopek

Osijek, 2019.

Sadržaj

1. Uvod

U projektnom zadatku proučena je Fourierova transformacija i njene dvije izvedbe, diskretna i brza Fourierova transformaciju. Na osnovu teorija implementirano je programsko rješenje koje uklanja šumove u slikama. Svaka vrsta signala, bilo da se radi o komunikaciji, audio zapisu ili digitalnoj fotografiji, se početno nalazi u vremenskoj domeni što nije pogodno za obradu istog. Upravo zbog toga se koristi Fourierova transformacija čiji je zadatak rastaviti signal na njegove frekvencijske komponente te na taj način omogućiti obradu signala. Početna ideja matematičara Jean Baptiste Joseph Fouriera je bila da se svaka periodična funkcija može zapisati kao suma sinusa različitih amplituda, faza i frekvencija. Na osnovu te teorije su nastale diskretna Fourierova transformacija i brza Fourierova transformacija koje će se objasniti u ovom projektnom zadatku.

2. Fourierova transformacija

Fourierova transformacija rastavlja bilo koji oblik funkcije u zbroj sinusoidnih temeljnih funkcija od kojih je svaka složena od različitih frekvencija. Onoga trenutka kada je francuski matematičar Jean Baptiste Joseph Fourier došao do tog zaključka, otvorili su se novi pogledi kako se funkcija ponaša i koje se sve potencijalne ideje mogu ostvariti ovim zaključkom. Ono što ovaj oblik transformacije čini posebnim je činjenica da omogućuje pojednostavljenje kompliciranih problema, ali još bitnije, omogućeno je rješenje problema koji do tada nisu bili matematički rješivi.

Fourierova transformacija funkcije *g(t)* je definirana kao:

$$F\{g(t)\} = G(f) = \int_{-\infty}^{\infty} g(t)e^{-i2\pi ft}\, dt$$

gdje je *i* kompleksni broj za kojeg vrijedi da je $i^2 = -1$

Rješenje integrala je funkcija *f*, a na osnovu toga *G(f)* prikazuje koliko snage *g(t)* sadrži u trenutku frekvencije *f*. Također je korištenjem inverzne Fourierove transformacije moguće odrediti obrnuti proces, odnosno, prebacivanje funkcije *G(f)* iz frekvencijske domene u funkciju *g(t)* vremenske domene:

$$\mathcal{F}^{-1}\{G(f)\} = \int_{-\infty}^{\infty} G(f)e^{i2\pi ft}\, df = g(t)$$

Za definiranje Fourierove transformacije nužan je dovoljan uvjet, odnosno, apsolutna integrabilnost *x(t)* i potreban uvjet koji su zapravo Dirichletovi uvjeti koji kažu da interval u kome je funkcija definirana, može se podijeliti u podintervale tako da je u svakom od njih funkcija neprekinuta i monotona te da u svakoj točki prekidnosti postoji *x(t+0)* i *x(t-0)*.

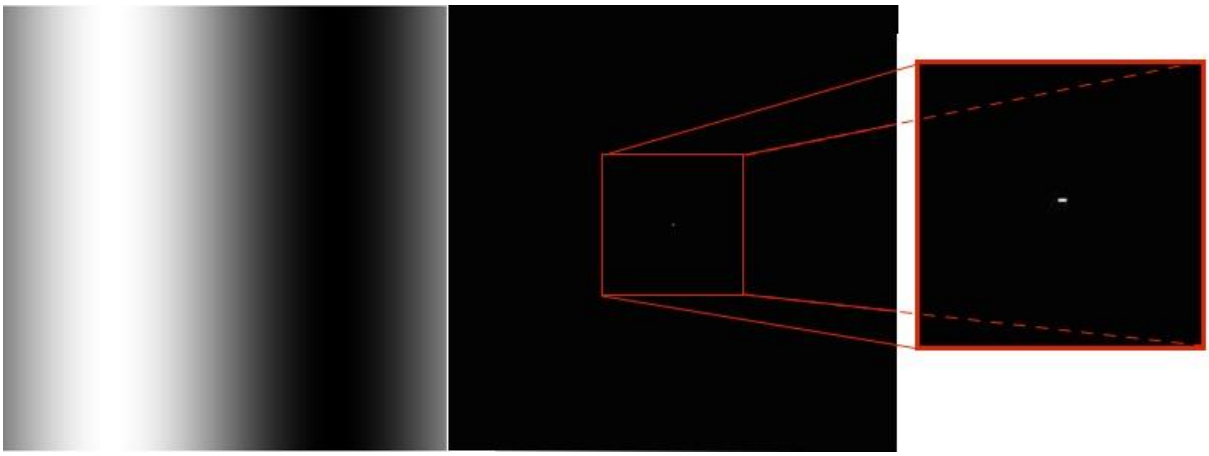
Svaku digitalnu sliku možemo promatrati kao promjenjivu funkciju, ali umjesto da se takav signal mijenja u vremenskoj domeni, on se mijenja u dvodimenzionalnoj prostornoj domeni. U sivoj digitalnoj slici svaki od piksela ima vrijednost između 0 i 255 što predstavlja intenzitet svjetline piksela. Slike se također mogu izraziti kao zbroj sinusoida u dvodimenzionalnoj prostornoj domeni i mogu se zapisati kao:

$$z = a\sin(hx + ky)$$

gdje *x* i *y* predstavljaju kordinate točaka, *z* je visina ili intenzitet sinusnog vala u određenoj točki, *a* je amplituda (maksimalna visina *z*), dok *h* i *k* predstavljaju broj koliko puta se sinusni val ponavlja u *x* i *y* smjeru, odnosno predstavljaju frekvenciju *x* i *y* koordinata.

Svaki piksel u Fourierovoj transformaciji ima koordinatu *h* i *k* koja predstavlja doprinos sinusoidnog vala s frekvencijom *x* i *y*. Središnja točka (centar) koju možemo zapisati sa koordinatama (0,0) i njen intenzitet svjetline je srednja vrijednost piksela na slici. Točke koje se nalaze lijevo i desno od središta predstavljaju sinusoidne valove koji se protežu po *x* osi (*k=0*), a intenzitet svjetline tih točaka predstavlja frekvenciju u Fourierovoj transformaciji.

Vertikalne točke naspram središta predstavljaju sinusoidne valove koji se protežu po *y* osi, ali vrijednost im ostaje konstantna u *x* (*h=0*). Primjer je prikazan na slici 1.



Slika 1. Signal *sin(x)* (grayscale) i njegova Fourierova transformacija

Slika 1. predstavlja sliku dvodimenzionalnog sinusoidnog signala *sin(x)*, a desno je njegova Fourierova transformacije. Druga slika ima iste dimenzije kao i prva te je potpuno crna osim tri piksela koja su bijele boje. Jedan piksel (središnji) sa koordinatama (0,0) i druga dva piksela lijevo i desno od središnjeg sa koordinatama (-1,0) i (1,0).

Fourierova transformacija jednostavnog signala ima vrlo malo piksela koji su svijetlih intenziteta (bijele boje), ali u slučaju kompleksnih digitalnih slika, postoji mnogo više piksela svijetlih intenziteta jer je takva slika opisana s više sinusoidnih valova. Naglašeni intenzitet piksela duž *x* i *y* osi Fourierove transformacije se može povezati s prirodom oko nas gdje se nalazi mnogo horizontalnih i okomitih obilježja poput zidova, ravnih površina pa čak i tijela ljudi, životinja i biljaka koja su simetrična oko okomite osi.

3. Definicija i vrste korištenih šumova

Šumovi u slikama su slučajne varijacije svjetline ili boja, odnosno nepoželjan rezultat snimanja digitalnim uređajem kojim se uklanjaju željene informacije u slici. Količina šumova se povećava ovisno o duljini ekspozicije, temperaturi i osjetljivosti digitalnog uređaja.

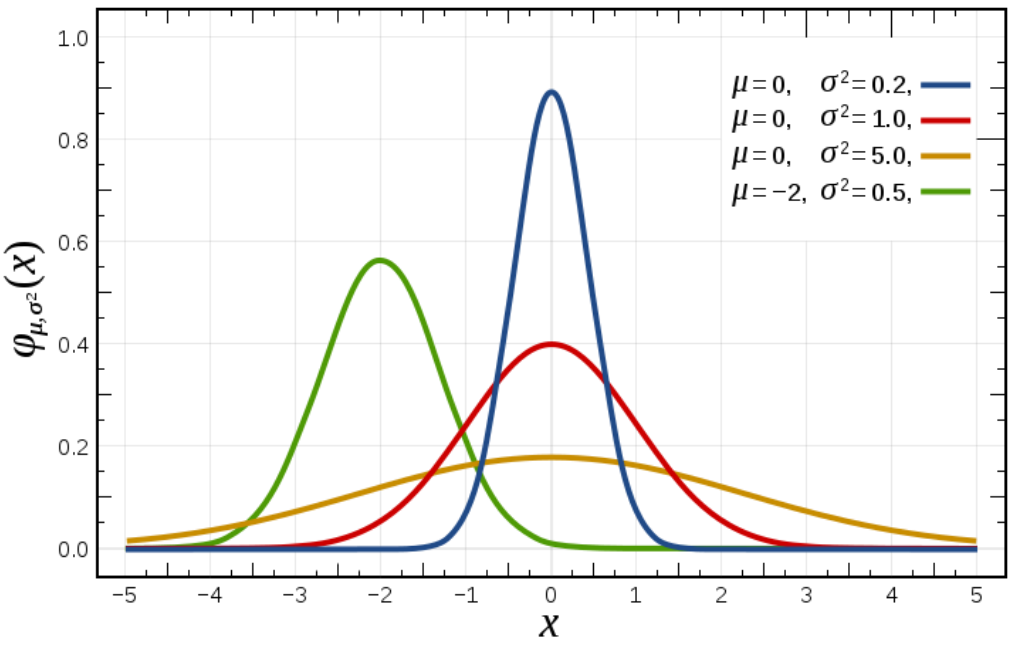
U radu su korištene slike koje sadrže različiti vrste šumova te se na taj način može zaključiti i primijetiti na koji način diskretna i brza Fourierova transformacija uklanjaju iste.

3.1 Gaussov šum

Gaussov šum je tipičan šum koji je neovisan na svakom pikselu te je neovisan o intenzitetu signala. Takav šum predstavlja statistički šum u kojem je vjerojatnost pojavljivanja određene vrijednosti jednaka normalnoj tj. Gaussovoj distribuciji. Vrijednost pojavljivanja neke slučajne vrijednosti z se može zapisati kao:

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

gdje je z razina sive boje, μ je srednja vrijednost, dok je σ standardna devijacija. Gaussova distribucija za određene parametre μ i σ prikazana je na slici 2.



Slika 2. Gaussova distribucija za različite parametre μ i σ

Za potrebe ovog projekta korištena je testna slika s Gaussovim šumom gdje je srednja vrijednost $\mu=0$, a standardna devijacija $\sigma=60$.

3.2 Uniformni šum

Uniformni šum koji se još naziva i kvantizacijskim šumom najčešće nastaje pri kvantiziranju piksela ulazne slike na određeni broj diskretnih razina. Pošto ima uniformnu razdiobu, to znači da svaka vrijednost unutar nekog raspona ima jednaku vjerojatnost pojavljivanja. Može se zapisati kao:

$$f(x) = \begin{cases} \frac{1}{b-a} \; za \; a \; \leq x \; \leq b, \\ 0 \; za \; x \; < a \; ili \; x \; > b \end{cases}$$

gdje su *a* i *b* granice unutar kojih se vrijednost može pojaviti.

Za potrebe ovog projekta korištena je testna slika s uniformnim šumom gdje su definirane granice *a=-60* i *b=60*.

3.2 Salt and pepper šum

Salt and pepper šum je oblik šuma gdje je neki postotak slučajnih piksela na slici ili crn ili bijel, odnosno slika će sadržavati crne piksele u svijetlim područjima i bijele piksele u tamnim područjima.

Za potrebe ovog projekta korištena je testna slika s *salt and pepper* šumom gdje je postotak slučajnih (crnih i bijelih) piksela 15%.

4. Diskretna Fourierova transformacija

Diskretna Fourierova transformacija (*engl. Discrete Fourier Transformation*) je uzorkovana Fourierova transformacija te već samim time ne može sadržavati sve frekvencijske komponente koje formiraju slike, već sadrži samo skup uzoraka koji je dovoljno velik da u potpunosti opiše sliku u prostornoj domeni. Broj frekvencija odgovara broju piksela u slici koja se nalazi u prostornoj domeni, tj. slika u prostornoj i Fourierovoj domeni je iste veličine.

Za sliku dimenzija *N x N*, diskretna Fourierova transformacija se može zapisati kao:

$$F(k,l) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(i,j)e^{-i2\pi(\frac{k_i}{N}+\frac{l_j}{N})}$$

gdje *f(i,j)* je slika u prostornoj domeni, a eksponencijalni dio je funkcija koja odgovara svakoj točki *F(k,l)*. Izraz se također može izraziti kao da vrijednost svake točke *F(k,l*) se dobiva množenjem slike u prostornoj domeni s odgovarajućom osnovnom funkcijom i zbrajanjem rezultata. Osnovne funkcije su sinusoidni valovi s frekvencijom koja se povećava, tj. predstavlja DC komponentu slike koja odgovara srednjoj vrijednosti svjetline gdje *F(N-1, N-1)* predstavlja najveću frekvenciju.

Na sličan način je sliku u Fourierovoj domeni moguće vratiti u prostornu domenu korištenjem inverzne formule:

$$f(i,j) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} F(k,l)e^{i2\pi(\frac{k_i}{N}+\frac{l_j}{N})}$$

Da bi se dobio navedeni rezultat inverzne Fourierove transformacije, potrebno je izračunati dvostruku sumu za svaku točku slike, a s obzirom na to da je Fourierova transformacija sama po sebi separabilna, može se zapisati:

$$F(k,l) = \frac{1}{N}\sum_{j=0}^{N-1} P(k,j)e^{-i2\pi\frac{l_j}{N}}$$

gdje je *P(k,j)*:

$$P(k,j) = \frac{1}{N}\sum_{i=0}^{N-1} f(i,j)e^{-i2\pi\frac{k_i}{N}}$$

Korištenjem ove dvije formule, slika u prostornoj domeni se najprije prebacuje u međufaznu (*engl. intermediate*) sliku pomoću *N* jednodimenzionalnih Fourierovih transformacija te se nakon toga prebacuje u konačnu sliku ponovnim korištenjem *N* jednodimenzionalne Fourierove

transformacije. Ovaj način izražavanja dvodimenzionalne Fourierove transformacije u smislu niza dvije jednodimenzionalne transformacije, smanjuje broj potrebnog računanja no usprkos tome, jednodimenzionalna diskretna Fourierova transformacija ima složenost $O(N^2)$. Složenost se može smanjiti na $O(N \log_2 N)$ tako da se koristi brza Fourierova transformacija za računanje jednodimenzionalne diskretne Fourierove transformacije što uvelike donosi poboljšanja, pogotovo za velike slike.

4.1 Implementacija diskretne Fourierove transformacije

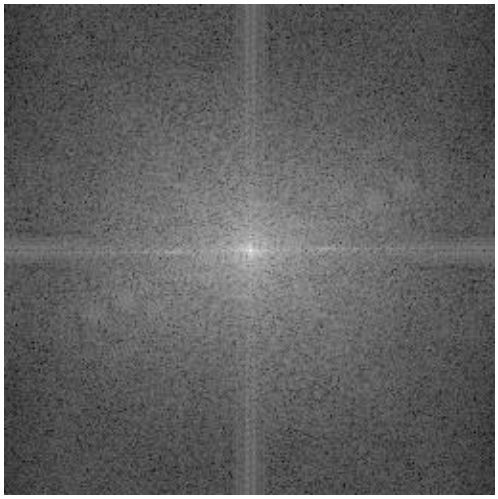
U osnovi, Fourierovu transformaciju koristimo ako želimo pristupiti geometrijskim karakteristikama slike koja se nalazi u prostornoj domeni. Budući da je slika u Fourierovoj domeni rastavljena na svoje sinusoidne komponente, lako je ispitati i obraditi određene frekvencije slike i na taj način utjecati na geometrijsku strukturu prostorne domene.

U većini implementacija korištenjem Fourierove transformacije, slika se pomiče na takav način da je DC komponenta, odnosno srednja vrijednost slike pomaknuta u središte *F(0,0)*. Što je slika dalje od središta to je veća njena odgovarajuća frekvencija.

Na primjeru ulazne slike, primjenjena je Fourierova transformacija.

Slika 3. Ulazna slika i rezultat Fourierove transformacije

Iz primjera se može vidjeti da su DC komponente najveće vrijednosti ulazne slike, ali je dinamički raspon Fourierovih koeficijenata prevelik, te se zbog toga sve ostale vrijednosti prikazuju kao pikseli crne boje. Zbog toga se primjenjuje logaritamska transformacija, nakon koje se dobije sljedeći rezultat:

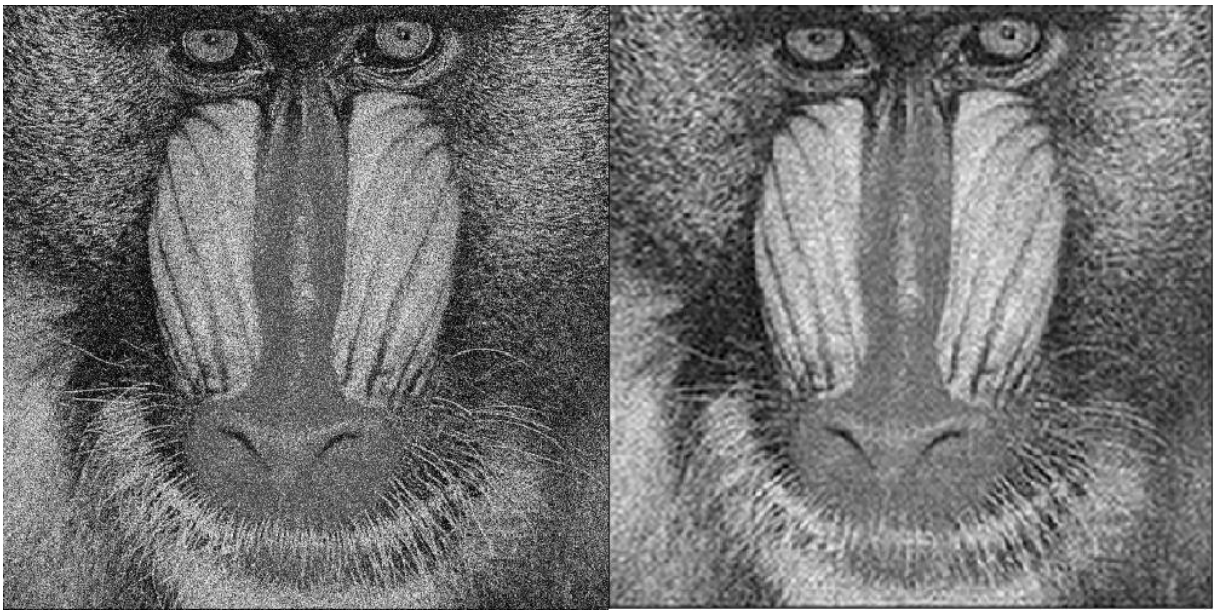


Slika 4. Fourierova transformacija (spektar) nakon logaritmaske transformacije

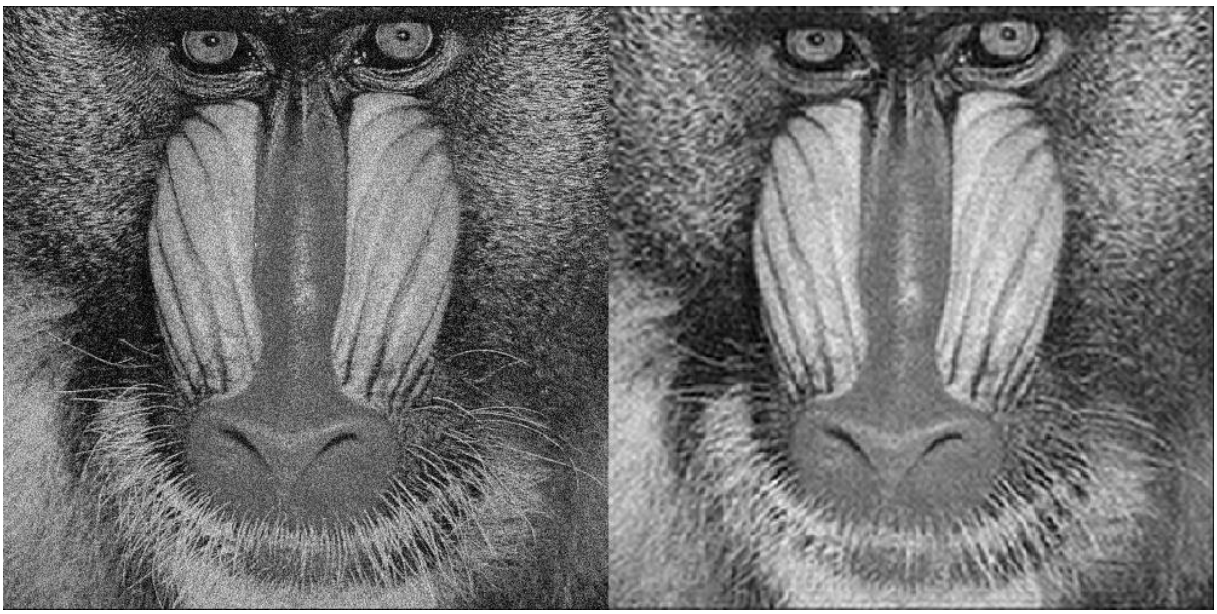
Iz rezultata se može zaključiti da slika sadrži komponente svih frekvencija, ali njihova veličina postaje manja za više frekvencije. Stoga, niske frekvencije sadrže više informacija o slici nego one više frekvencije. Rezultat također prikazuje da postoje dva dominantna smjera u slici Fourierove transformacije, jedan koji prolazi vertikalno, a drugi horizontalno kroz središte, kao što je spomenuto u prošlom poglavlju.

Nakon što su dobiveni rezultati slike korištenjem diskretne Fourierove transformacije kreirana je maska gdje središnji piksel ima vrijednost 1, odnosno bijele je boje, dok svi ostali imaju vrijednost 0, odnosno crne su boje. Nad umnoškom takve maske i Fourierovog spektra ulazne slike obavljena je inverzna funkcija koja će pomaknuti sve komponente slike s vrijednosti 0 u središte spektra. U slučaju kada se bilo koja vrsta šuma nalazi na slici, inverzna diskretna Fourierova transformacija će nad dobivenim rješenjem rezultirati slikom u kojoj su šumovi značajno ublaženi.

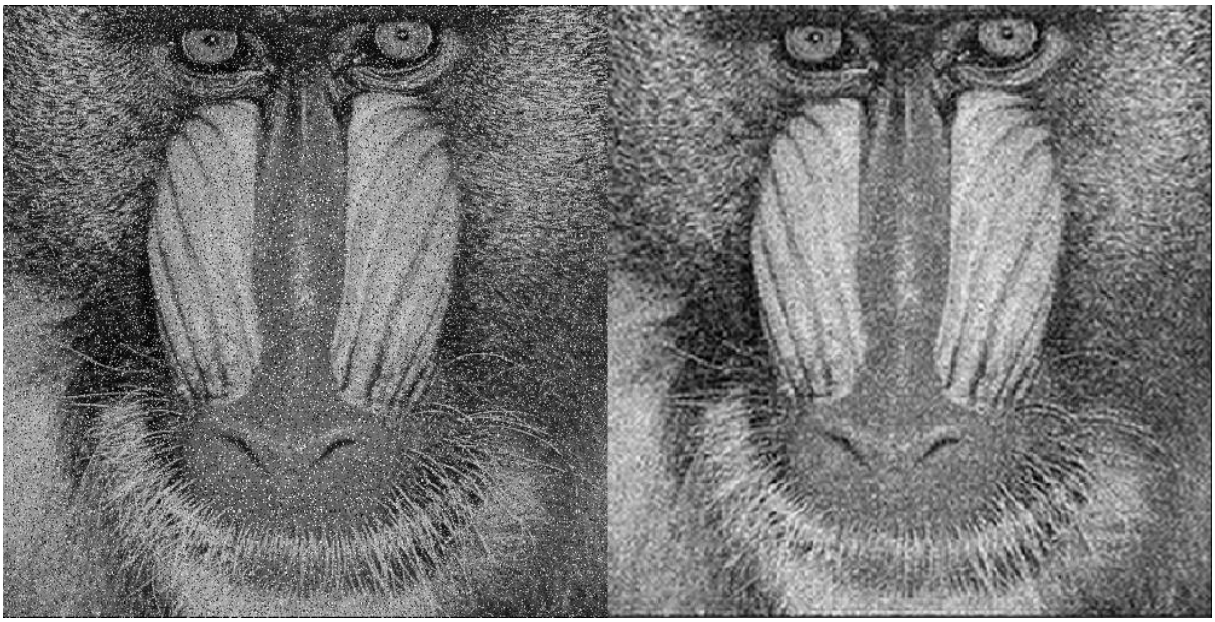
4.2 Rezultati uklanjanja šumova korištenjem DFT



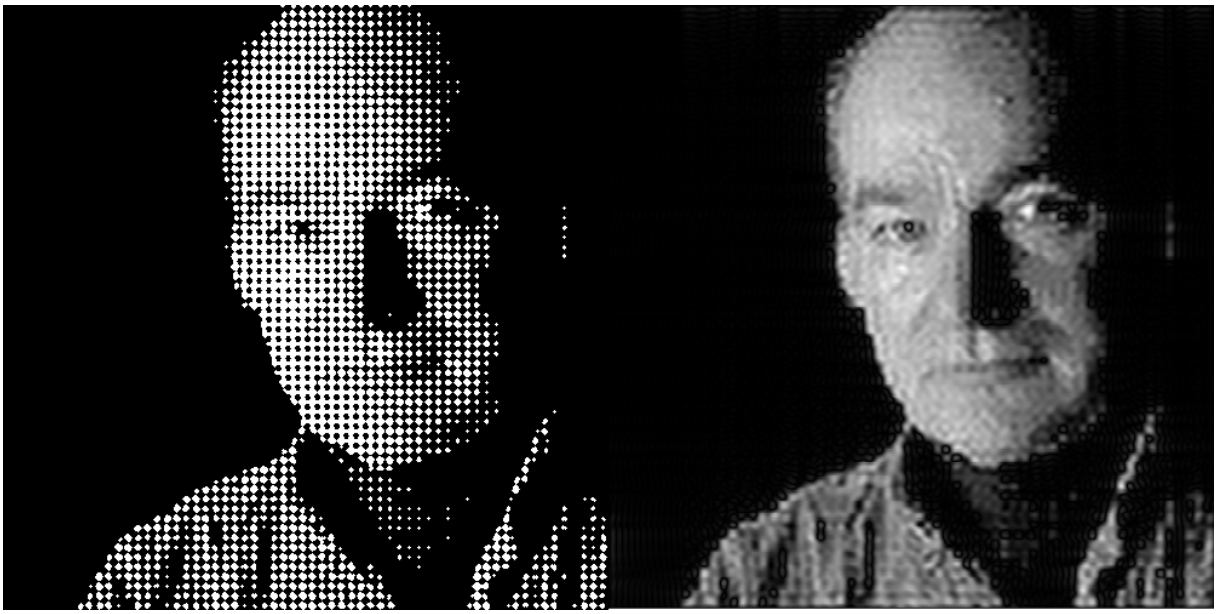
Slika 5. Izvorna slika (Gaussov šum) i obrađena slika



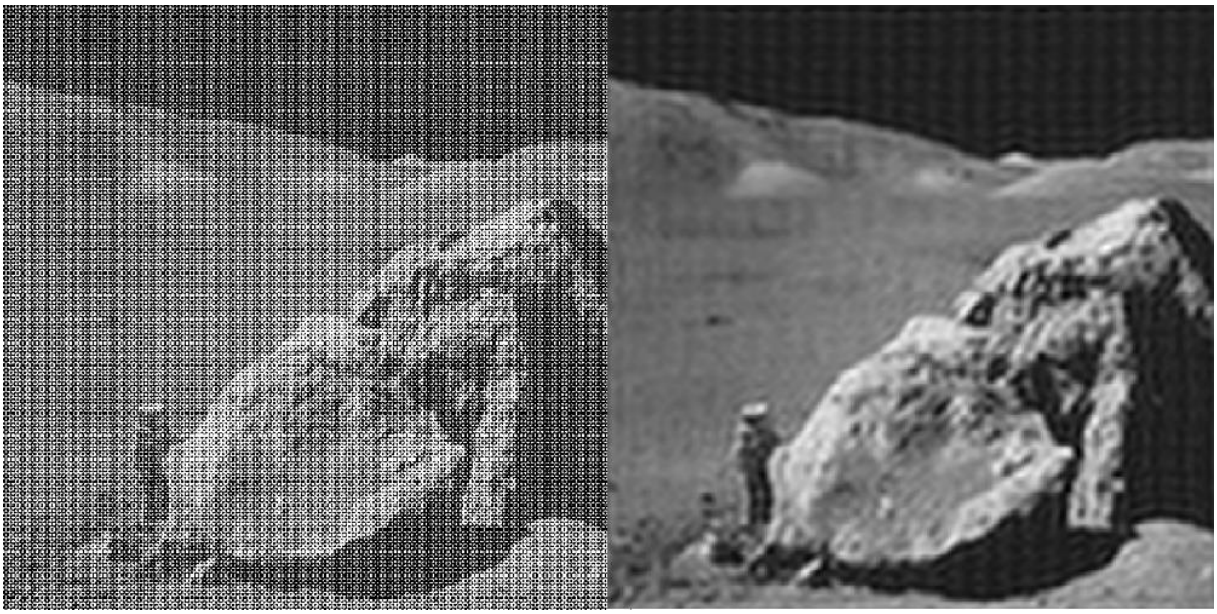
Slika 6. Izvorna slika (uniformni šum) i obrađena slika



Slika 7. Izvorna slika (*salt and pepper* šum) i obrađena



Slika 8. Izvorna slika (bijeli šum) i obrađena slika



Slika 8. Izvorna slika (sinusni visokofrekventni šum) i obrađena slika

5. Brza Fourierova transformacija

Brza Fourierova transformacija (*engl. Fast Fourier Transformation*) je učinkovita implementacija diskretne Fourierove transformacije koja se primjenjuje za pretvaranje slike iz prostorne domene u frekvencijsku domenu. Kada se slika prebaci u frekvencijsku domenu, moguće je primijeniti filtre pomoću konvolucija. Nakon toga je moguće primijeniti inverznu transformaciju u frekvencijskoj domeni kako bi se dobio rezultat konvolucije. Upravo se brza Fourierova transformacija koristi kako bi se računalno zahtjevne operacije konvolucije izbjegle te se koriste jednostavnije operacije množenja te je kao rezultat toga manja složenost računanja $[O(N \log_2 N)]$ od diskretne Fourierove transformacije $[\mathrm{O}(\mathrm{N}^2)]$. Razlika u brzini može biti ogromna, posebno za duge skupove podataka gdje *N* vrijednost može biti reda 10^6. Također, u slučaju pogreške kod zaokruživanja vrijednosti, brza Fourierova transformacija ima točnije rezultate od diskretne Fourierove transformacije.

Brza Fourierova transformacija i njena inverzna formula dvodimenzionalne slike se može zapisati kao:

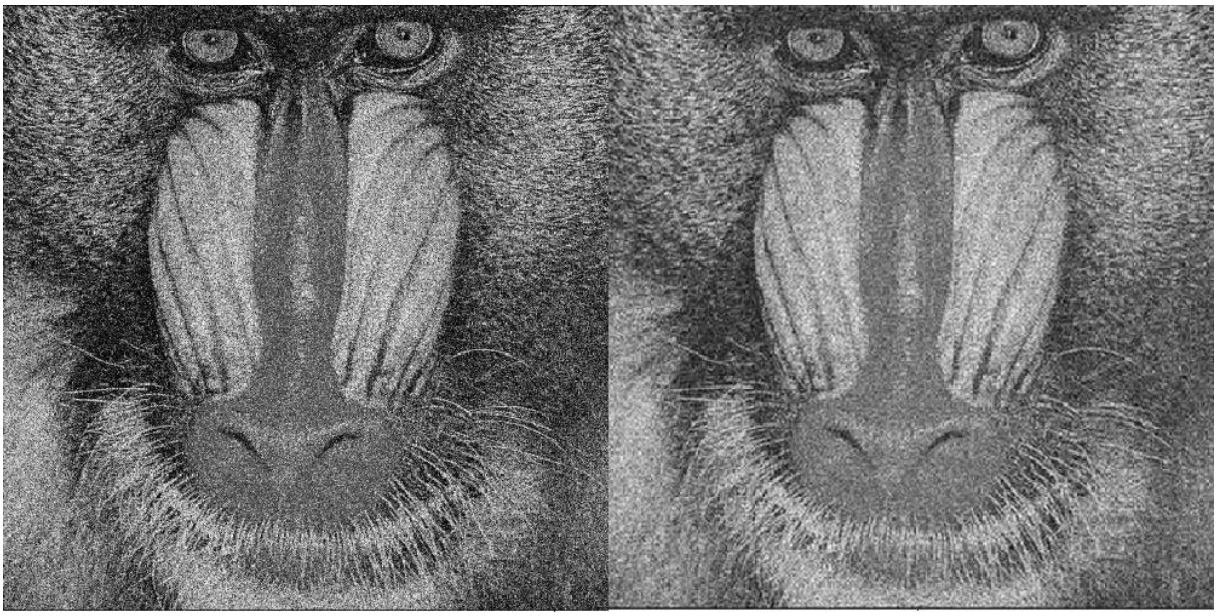
$$F(x) = \sum_{n=0}^{N-1} f(n) e^{-j2\pi(x\frac{n}{N})}$$

$$F(n) = \frac{1}{N} \sum_{n=0}^{N-1} F(x) e^{j2\pi(x\frac{n}{N})}$$

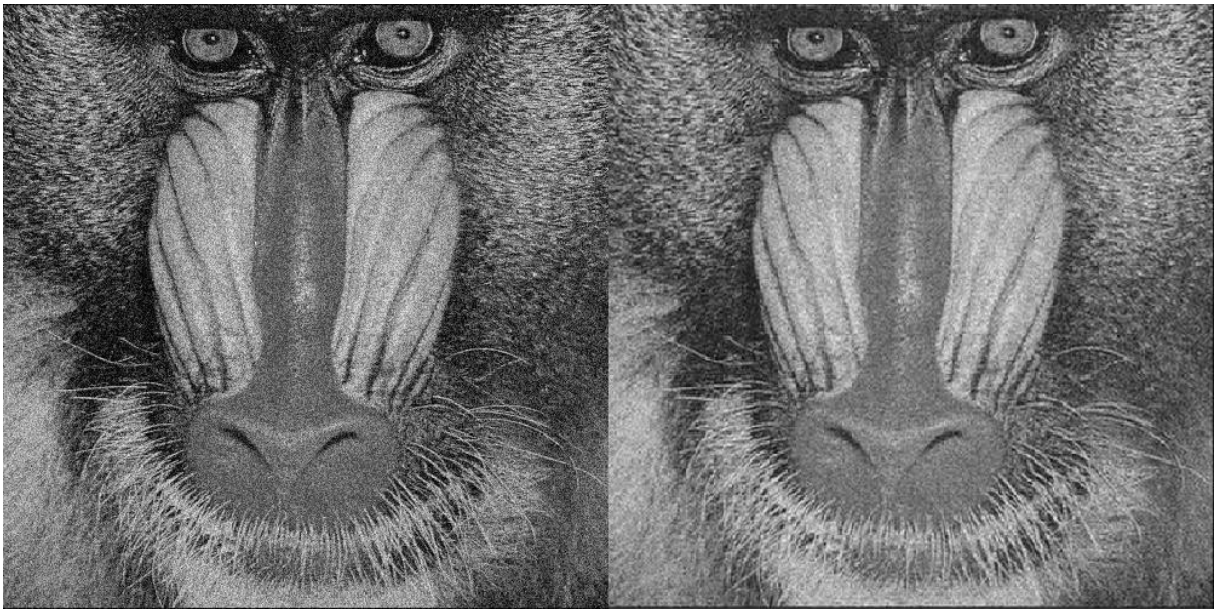
gdje *f(m,n)* predstavlja piksele na koordinatama *m* i *n*, *F(x,y)* je vrijednost slike u frekvencijskoj domeni koja odgovara koordinatama *x* i *y*, a *M* i *N* su dimenzije slike.

Zanimljivo svojstvo brze Fourierove transformacije je da se transformacija *N* točaka može prepisati kao zbroj dvije $\frac{N}{2}$ transformacije. Ovo svojstvo je bitno jer se neka računanja mogu ponovo iskoristiti čime se eliminiraju skupe operacije. Rješenje brze Fourierove transformacije je kompleksni broj koji ima mnogo veći raspon vrijednosti od slike koja se nalazi u prostornoj domeni te se zbog toga vrijednosti pohranjuju u *float* kako bi se točno mogle prikazati.

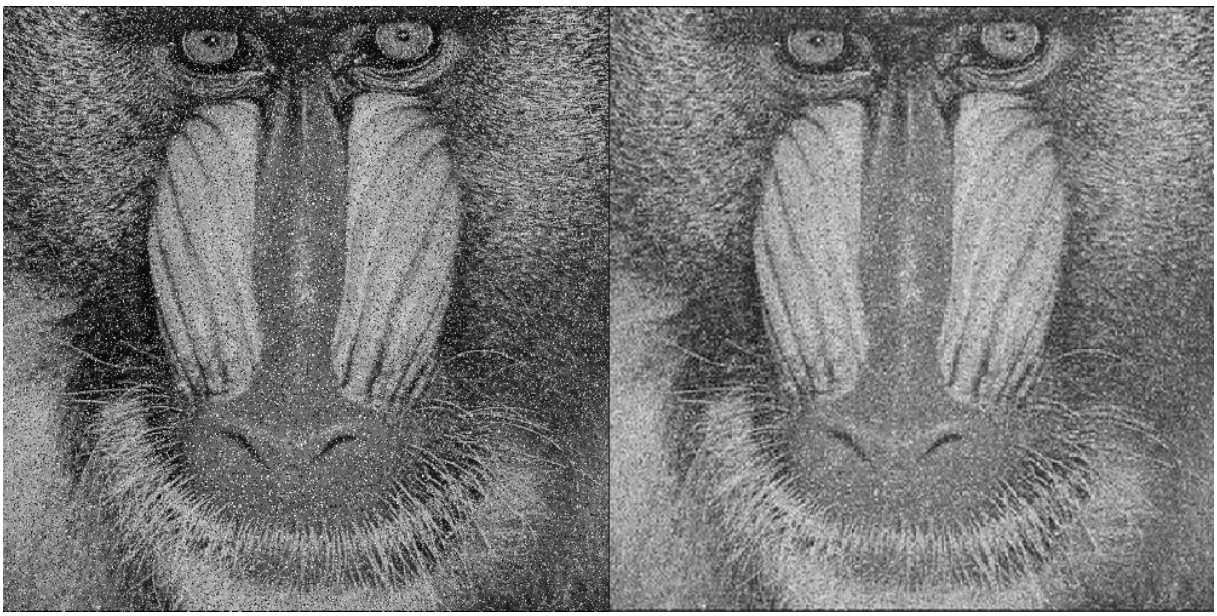
5.1 Rezultati uklanjanja šumova korištenjem FFT



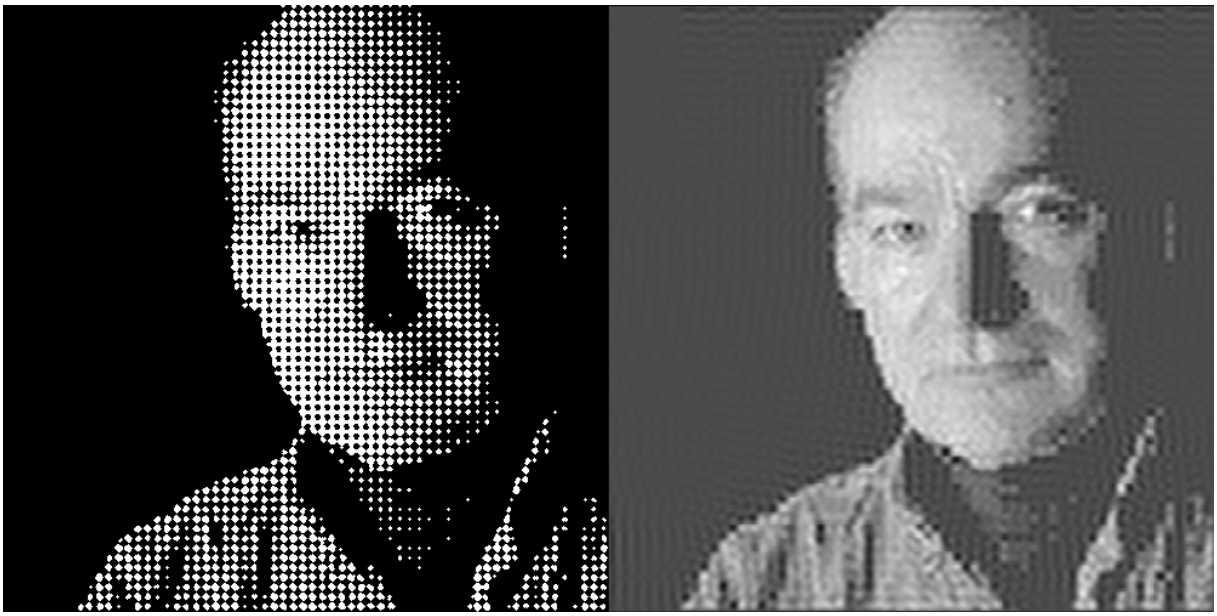
Slika 9. Izvorna slika (Gaussov šum) i obrađena slika



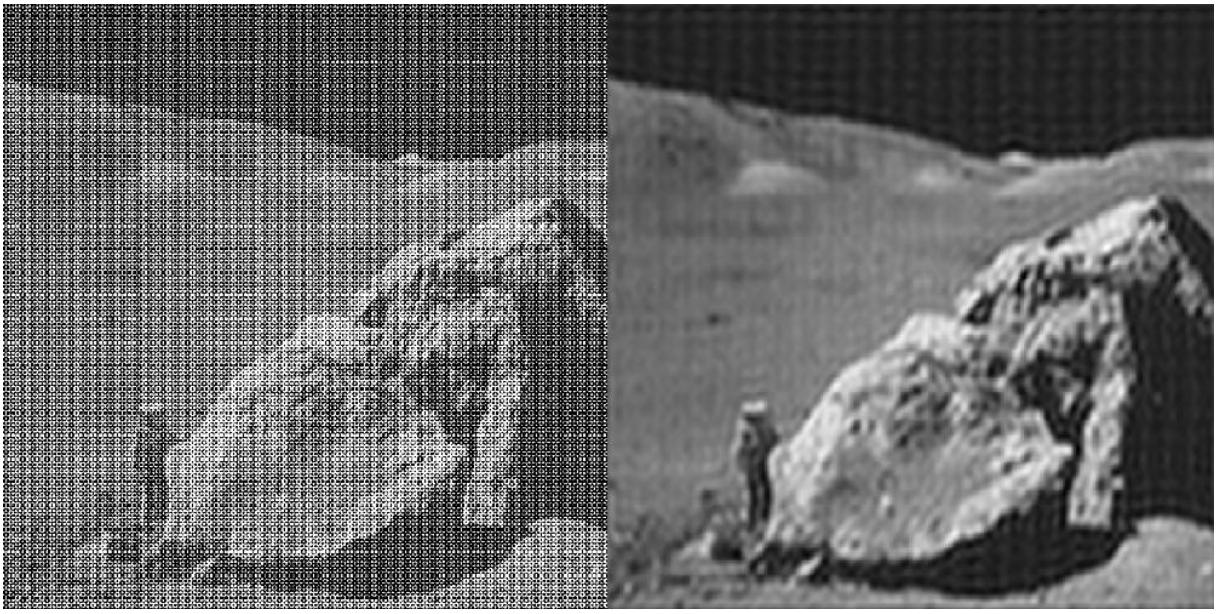
Slika 10 Izvorna slika (uniformni šum) i obrađena slika



Slika 11. Izvorna slika (*salt and pepper* šum) i obrađena slika



Slika 12. Izvorna slika (bijeli šum) i obrađena slika



Slika 12. Izvorna slika (sinusni visokofrekventni šum) i obrađena slika

Zaključak

U projektnom zadatku su obrađene diskretna i brza Fourierova transformacija i implementiranjem metode nad slikama s različitim vrstama šumova su dobiveni rezultati. Također, osim drugačijih vrsta šumova, ulazne slike u bile različitih *NxM* dimenzija, od najmanje slike (bijeli šum) 225x240 piksela, do najveće slike (visokofrekventni sinusni šum) dimenzija 630x474 piksela. Korištenjem *time()* funkcije dolazi se do zaključka da je brza Fourierova transformacija deset puta brža (*404ms*) od diskretne Fourierove transformacije (*47ms*), u obzir mjerenja vremena izvršavanja je uzeta slika s najvećim dimenzijama i najzahtjevnijim šumom za obradu. Glavne značajke brze Fourierove transformacije, brzina i učinkovitost su dokazane te osim bržeg algoritma se kvaliteta obrađene slike znatno razlikuje naspram diskretne Fourierove transformacije.

Također se dolazi do zaključka da korištenjem drugačijih tipova maski koje se primjenjuju na spektar Fourierove transformacije i tako rezultiraju obrađenom slikom, mogu se dobiti bolja rješenja od trenutnih. Primjer je slika s visokofrekventnim sinusnim šumom nad čijim se Fourierovim spektrom primjenjuje maska čiji je središnji piksel bijele, a svi ostali su crne boje. Rezultati su zadovoljavajući i glavni zadatak je odrađen, ali algoritam bi odradio puno bolji posao da se na primjer koristi maska u obliku kružnice čiji su pikseli crne boje, a svi ostali su bijele boje.

Literatura

http://www.thefouriertransform.com/transform/properties.php

https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm

https://software.intel.com/en-us/articles/implementation-of-fast-fourier-transform-for-image-processing-in-directx-10

http://www.differencebetween.net/technology/difference-between-fft-and-dft/

https://en.wikipedia.org/wiki/Image_noise

https://mathematica.stackexchange.com/questions/110914/how-to-use-2d-fourier-analysis-to-clean-the-noise-in-an-image

Programsko riješenje

Algoritam za uklanjanje šumova pomoću diskretne Fourierove transformacije

```
1. import numpy as np
2. import cv2
3. from matplotlib import pyplot as plt
4.
5. #Load source image
6. image_source = plt.imread('C:/FaksGit/FourierFilter/TestImages/baboon_gauss60.png').
   astype(float)
7. img_float32 = np.float32(image_source)
8.
9. #Plot input image
10. plt.figure()
11. plt.imshow(image_source, plt.cm.gray)
12. plt.xticks([]), plt.yticks([])
13. plt.title('Original image')
14.
15. #Performs a forward or inverse Discrete Fourier transform of a 1D or 2D floating-
   point array
16. dft = cv2.dft(img_float32, flags = cv2.DFT_COMPLEX_OUTPUT)
17. #Shift the zero-frequency component to the center of the spectrum.
18. dft_shift = np.fft.fftshift(dft)
19.
20. rows, cols = image_source.shape
21. crow, ccol = rows/2 , cols/2  # centar
22.
23. keep_fraction = 60
24.
25. #create a mask first, center square is 1, remaining all zeros
26. mask = np.zeros((rows, cols, 2), np.uint8)
27. mask[int(crow)-keep_fraction:int(crow)+keep_fraction,
28. int(ccol)-keep_fraction:int(ccol)+keep_fraction] = 1
29.
30. # apply mask and inverse DFT
31. fshift = dft_shift * mask
32.
33. #The inverse of fftshift. Although identical for even-length x,
34. #the functions differ by one sample for odd-length x
35. f_ishift = np.fft.ifftshift(fshift)
36.
37. #Calculates the inverse Discrete Fourier Transform of a 1D or 2D array.
38. img_back = cv2.idft(f_ishift)
39.
40. #Calculates the magnitude of 2D vectors
41. img_back = cv2.magnitude(img_back[:,:,0],img_back[:,:,1])
42.
43. fig = plt.figure(figsize=(20,10))
44. plt.imshow(img_back, plt.cm.gray)
45. plt.xticks([]), plt.yticks([])
46. plt.title('Reconstructed Image DFT')
47. plt.show()
48. fig.savefig('baboon_gauss60_DFT60.png')
```

Algoritam za uklanjanje šumova pomoću brze Fourierove transformacije

```
1. import numpy as np
2. import matplotlib.pyplot as plt
3. from scipy import fftpack
4. from matplotlib.colors import LogNorm
5. import cv2
6.
7. #Load input image
8. image_source = cv2.imread('C:/FaksGit/FourierFilter/TestImages/baboon_gauss60.png')
9. gray_image = cv2.cvtColor(image_source, cv2.COLOR_BGR2GRAY)
10.
11. #Plot input image
12. plt.figure()
13. plt.imshow(gray_image, plt.cm.gray)
14. plt.xticks([]), plt.yticks([])
15. plt.title("Original image")
16.
17. #Return the two-dimensional discrete Fourier transform of the 2-D argument x.
18. image_fft = fftpack.fft2(gray_image)
19.
20. #Logaritmic map
21. def show_spectrum(image_fft):
22.     plt.imshow(np.abs(image_fft), norm=LogNorm(vmin=5))
23.     plt.colorbar()  #Add colorbar
24.
25. #Plot FT input image
26. plt.figure()
27. show_spectrum(image_fft)
28. plt.title("Fourier transform")
29.
30. keep_fraction = 0.3  #keep fraction (u oba smijera)
31. image_fft2 = image_fft.copy()
32. row, col = image_fft2.shape  #get the current shape of an array
33.
34. #Set on zero all rows with index between row*keep_fraction and
35. row*(1-keep_fraction)
36. image_fft2[int(row*keep_fraction):int(row*(1-keep_fraction))] = 0
37. #Similar for columns
38. image_fft2[:, int(col*keep_fraction):int(col*(1-keep_fraction))] = 0
39.
40. plt.figure()
41. show_spectrum(image_fft2)
42. plt.title('Filtered Spectrum')
43.
44. #Return inverse two-dimensional discrete Fourier transform
45. of arbitrary type sequence x
46. image_new = fftpack.ifft2(image_fft2).real
47.
48. fig = plt.figure(figsize=(20,10))
49. plt.imshow(image_new, plt.cm.gray)
50. plt.xticks([]), plt.yticks([])
51. plt.title('Reconstructed Image FFT')
52. fig.savefig('baboon_gauss60.3FFT.png')
```